import pandas as pd

import numpy as np

from sklearn.model\_selection import train\_test\_split

from sklearn.feature\_extraction.text import TfidfVectorizer

from sklearn.linear\_model import LogisticRegression

from sklearn.metrics import accuracy\_score, confusion\_matrix, roc\_curve, roc\_auc\_score

import nltk

from nltk.corpus import stopwords

from collections import Counter

import matplotlib.pyplot as plt

import seaborn as sns

nltk.download('stopwords')

[nltk\_data] Downloading package stopwords to /usr/share/nltk\_data...

[nltk\_data] Package stopwords is already up-to-date!

*# Loading the dataset*

df = pd.read\_csv("/kaggle/input/sms-spam-collection-dataset/spam.csv",encoding='latin-1')

*# Displaying the first few rows of the dataset*

df.head()

*# Droping unnecessary columns from the DataFrame*

columns\_to\_drop = ["Unnamed: 2", "Unnamed: 3", "Unnamed: 4"]

df.drop(columns=columns\_to\_drop, inplace=True)

*# Displaying the data*

df

*# Consice information of the dataset*

df.info()

<class 'pandas.core.frame.DataFrame'>

RangeIndex: 5572 entries, 0 to 5571

Data columns (total 2 columns):

# Column Non-Null Count Dtype

--- ------ -------------- -----

0 v1 5572 non-null object

1 v2 5572 non-null object

dtypes: object(2)

memory usage: 87.2+ KB

*# Rename the columns "v1 and "v2" to new names*

new\_column\_names = {"v1":"Category","v2":"Message"}

df.rename(columns = new\_column\_names,inplace = True)

sns.countplot(data=df, x='Category')

plt.xlabel('Category')

plt.ylabel('count')

plt.title('Distribution of mails')

plt.show()

plt.pie(df['Category'].value\_counts(),labels=['ham','spam'],autopct='**%0.2f**')

plt.show()

df.loc[df["Category"] == "spam", "Category"] = 0

df.loc[df["Category"] == "ham", "Category"] = 1

*# Separate the feature (message) and target (category) data*

X = df["Message"]

Y = df["Category"]

print(X)

0 Go until jurong point, crazy.. Available only ...

1 Ok lar... Joking wif u oni...

2 Free entry in 2 a wkly comp to win FA Cup fina...

3 U dun say so early hor... U c already then say...

4 Nah I don't think he goes to usf, he lives aro...

...

5567 This is the 2nd time we have tried 2 contact u...

5568 Will Ì\_ b going to esplanade fr home?

5569 Pity, \* was in mood for that. So...any other s...

5570 The guy did some bitching but I acted like i'd...

5571 Rofl. Its true to its name

Name: Message, Length: 5572, dtype: object

print(Y)

0 1

1 1

2 0

3 1

4 1

..

5567 0

5568 1

5569 1

5570 1

5571 1

Name: Category, Length: 5572, dtype: object

X\_train, X\_test, Y\_train, Y\_test = train\_test\_split(X, Y, test\_size = 0.2, random\_state = 3)

In [22]:

*# Print the shape of X*

print(X.shape)

*# Print the shape of X\_train and X\_test*

print(X\_train.shape)

print(X\_test.shape)

*# Initialize TF-IDF Vectorizer*

feature\_extraction = TfidfVectorizer(min\_df=1, stop\_words="english", lowercase=True)

In [25]:

linkcode

*# Feature extraction for training and testing data*

X\_train\_features = feature\_extraction.fit\_transform(X\_train)

X\_test\_features = feature\_extraction.transform(X\_test)

In [26]:

*# Convert Y\_train and Y\_test to integer type*

Y\_train = Y\_train.astype("int")

Y\_test = Y\_test.astype("int")

print(X\_train)

3075 Mum, hope you are having a great day. Hoping t...

1787 Yes:)sura in sun tv.:)lol.

1614 Me sef dey laugh you. Meanwhile how's my darli...

4304 Yo come over carlos will be here soon

3266 Ok then i come n pick u at engin?

...

789 Gud mrng dear hav a nice day

968 Are you willing to go for aptitude class.

1667 So now my dad is gonna call after he gets out ...

3321 Ok darlin i supose it was ok i just worry too ...

1688 Nan sonathaya soladha. Why boss?

Name: Message, Length: 4457, dtype: object

*# Creating and Fit Logistic Regression Model*

model = LogisticRegression()

model.fit(X\_train\_features, Y\_train)

*#Make predictions on the training data*

predict\_train\_data=model.predict(X\_train\_features)

In [31]:

*#Model Evaluation*

from sklearn.metrics import accuracy\_score,confusion\_matrix

accuracy\_train\_data=accuracy\_score(Y\_train,predict\_train\_data)

print("Accuracy on training data: ",accuracy\_train\_data)

Accuracy on training data: 0.9661207089970832

In [32]:

linkcode

*# Make predictions on the testing data*

predict\_test\_data=model.predict(X\_test\_features)

*#Model Evaluation*

accuracy\_test\_data=accuracy\_score(Y\_test,predict\_test\_data)

print("acuuracy on test data: ",accuracy\_test\_data)

new\_mail=["Congratulations on your recent achievement! Well done."]

new\_data\_features=feature\_extraction.transform(new\_mail)

prediction=model.predict(new\_data\_features)

print(prediction)

if(prediction[0]==1):

print("Ham Mail")

else:

print("Spam Mail")

conf\_matrix=confusion\_matrix(Y\_test,predict\_test\_data)

plt.figure(figsize=(6,4))

sns.heatmap(conf\_matrix,annot=True,fmt="d",cmap="Oranges",cbar=False)

plt.xlabel("Predicted value")

plt.ylabel("Actual value")

plt.title("Confusion Matrix")

plt.show()

*# Data visualization - Top 7 Most Common Words in Spam Emails*

stop\_words = set(stopwords.words('english'))

spam\_words = " ".join(df[df['Category'] == 0]['Message']).split()

ham\_words = " ".join(df[df['Category'] == 1]['Message']).split()

spam\_word\_freq = Counter([word.lower() for word **in** spam\_words if word.lower() **not** **in** stop\_words **and** word.isalpha()])

plt.figure(figsize=(10, 6))

plt.bar(\*zip(\*spam\_word\_freq.most\_common(7)), color='y')

plt.xlabel('Words')

plt.ylabel('Frequency')

plt.title('Top 7 Most Common Words in Spam Emails')

plt.xticks(rotation=45)

plt.show()